Homework 8

The purpose of this homework is to practice examining unusual points in simple linear regression models and to work with multiple regression models. Please fill in the appropriate code and write answers to all questions in the answer sections, then submit a compiled pdf with your answers through gradescope by 11:59pm on Sunday November 10th.

As always, if you need help with any of the homework assignments, please attend the TA office hours which are listed on Canvas and/or ask questions on Piazza. Also, if you have completed the homework, please help others out by answering questions on Piazza.

Part 1: Preparing the data

Let's continue to examine Toyota Corolla data using the data set from Edmunds.com in order to gain practice examininig unusual points. Remember, the data has been made available to this class for educational purposes, so please do not share this data outside of the class.

Part 1.1 (5 points): Let's start again by loading the dplyr libary and the Edmunds data set using the code below. Again use the dplyr select() and filter() functions to create a reduced data frame object called `used_corollas_all`. However this time **do not do any filtering related to the number of miles a car has been driven** (i.e., keep in the data set cars that have been drive more than 150,000 miles).

In particular, follow the steps below:

1) The only variables that should be in the `used_corollas_all` data frame are:

a) model_bought: the model of the car
b) new_or_used_bought: whether a car was new or used when it was purchased
c) price_bought: the price the car was purchased for

2) The only cases should be in the `used_corollas_all` data frame are:

a) used cars
b) Toyota Corollas

3) Use the na.omit() function on the `used_corollas_all` data frame to remove cases that have missing values.

If you have properly filter the data, the `used_corollas_all` should have 249 cases, so check this is the case before going on to the next set of exercises.

Finally, recreate the used Corolla data set which you created in homework 7 that only includes cars that have been driven less than 150,000 miles, and save it in an object called `used_corollas_150k`. You should be able to create this data frame using just one line of R code once you have created the `used_corollas_all` data frame (and as you saw on homework 7, this data frame should have 248 cases).

```r
# load the dplyr library
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# load the data set
load("car_transactions.rda")

# use dplyr to created used_corollas_all that only has used Corollas
used_corollas_all <- filter(car_transactions, new_or_used_bought == "U",
                            make_bought == "Toyota", model_bought == "Corolla")
used_corollas_all <- select(used_corollas_all, model_bought, new_or_used_bought,
                            price_bought, mileage_bought)
used_corollas_all <- na.omit(used_corollas_all)


# check the size of the resulting data frame
dim(used_corollas_all)
```

```
## [1] 249   4
```

```r
# created the used_corollas_150k that contains only cars with less than 150,000 miles
used_corollas_150k <- used_corollas_all %>% filter(mileage_bought <= 150000)
dim(used_corollas_150k)
```

```
## [1] 248   4
```

Part 1.2 (10 points):

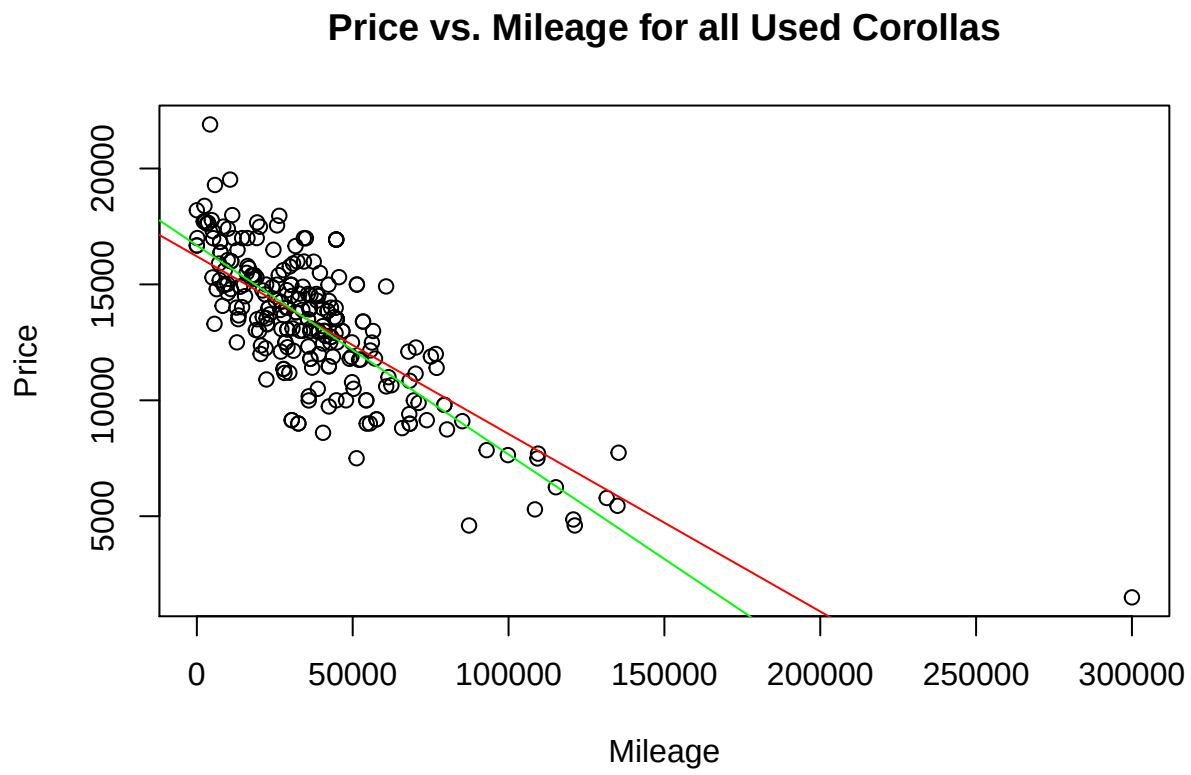
Now fit a linear regression model to the `used_corollas_all` that shows the predicted (expected) price as a function of the number of miles driven. Save this model to an object called `lm_fit_all`. Also, create a scatter plot of the price as a function of the number of miles driven, add a red line to our plot showing the regression line, and print the regression coefficients found.

Also, fit the regression model using the `used_corollas_150k` (as you did on homework 7). Add a green line to the plot showing the fit when the one additional data point is added, and describe below how similar the slope coefficient $\hat{\beta}_1$ is between these models. Finally, make a prediction for the price of a car that has been driven 150,000 miles using both the `lm_fit_all` and the `lm_fit_150k` models. Do these models seem similar to you?

```r
# start by plotting the data
plot(used_corollas_all$mileage_bought, used_corollas_all$price_bought,
     xlab = "Mileage", ylab = "Price", main = "Price vs. Mileage for all Used Corollas")


# fit a regression model, add the regression line to the plot and display the regression coefficients
lm_fit_all <- lm(price_bought ~ mileage_bought, data = used_corollas_all)
abline(lm_fit_all, col = "red")
coef(lm_fit_all)
```

```
##    (Intercept) mileage_bought
## 16210.25517005    -0.07660694
```

```r
# fit the model that is excluding the 150k data point
lm_fit_150k <- lm(price_bought ~ mileage_bought, data = used_corollas_150k)
abline(lm_fit_150k, col = "green")
```




```r
coef(lm_fit_150k)
```

```
##    (Intercept) mileage_bought
## 16681.91992781    -0.09018627
```

```r
# make a prediction for a car driven 150k miles using both models
predict(lm_fit_all, newdata = data.frame(mileage_bought = 150000))
```

```
##        1
## 4719.215
```

```r
predict(lm_fit_150k, newdata = data.frame(mileage_bought = 150000))
```

```
##        1
## 3153.979
```

Answers: $\hat{\beta}_1$ changes from -0.07660694 to -0.09018627 when the one point in excluded from the dataset. These two slope coefficients seem to be pretty similar but would result in large differences at higher mileages. At 150000 miles the data including the point predicts a price of $4719.215 while the data excluding the point predicts $3153.979. These results are pretty far apart so I would say the models are not very similar.

Part 1.3 (5 points): Create a 95% confidence interval for the value of the regression slope β_1 using the `used_corollas_150k`. If we were assuming that the confidence interval from the `used_corollas_150k` was reasonable, would the value for the regression slope found in the `used_corollas_all` model seem like a plausible value for what the true parameter value β_1 is (at the $\alpha = 0.05$ level)?

```r
(CI_150k <- confint(lm_fit_150k))
```

```
##                         2.5 %         97.5 %
## (Intercept)    16279.20570876 17084.63414685
## mileage_bought    -0.09912747    -0.08124508
```

Answer At $\alpha = 0.05$, the value for the regression slope found in `used_cororllas_all` would not be a plausible value for the true parameter since it is outside the confidence interval [-0.09912747, -0.08124508]. -0.0766 is not within this interval.

Part 1.4 (10 points): Now sort the data frame `used_corollas_all` so that the rows are in the order from smallest number of miles driven to the most number of miles driven, and store it again the same object called `used_corollas_all`. Refit the `lm_fit_all` using this sorted data frame (as a sanity check, the coefficients found should be the same as before). Then, recreate the scatter plot based on this sorted `used_corollas_all` data and add to this plot both the confidence intervals for **the regression line** in green and the prediction interval in blue (again using this sorted `used_corollas_all`). Describe how the equation for **the confidence interval for the regression line** leads to the fact that these confidence intervals becomes wider for cars that have been driven the most miles.

```r
# arrange the data and refit the model
used_corollas_all <- arrange(used_corollas_all, mileage_bought)
lm_fit_all <- lm(price_bought ~ mileage_bought, data = used_corollas_all)
coef(lm_fit_all) #We see these coefficients are the same as before
```

```
##    (Intercept) mileage_bought
## 16210.25517005    -0.07660694
```

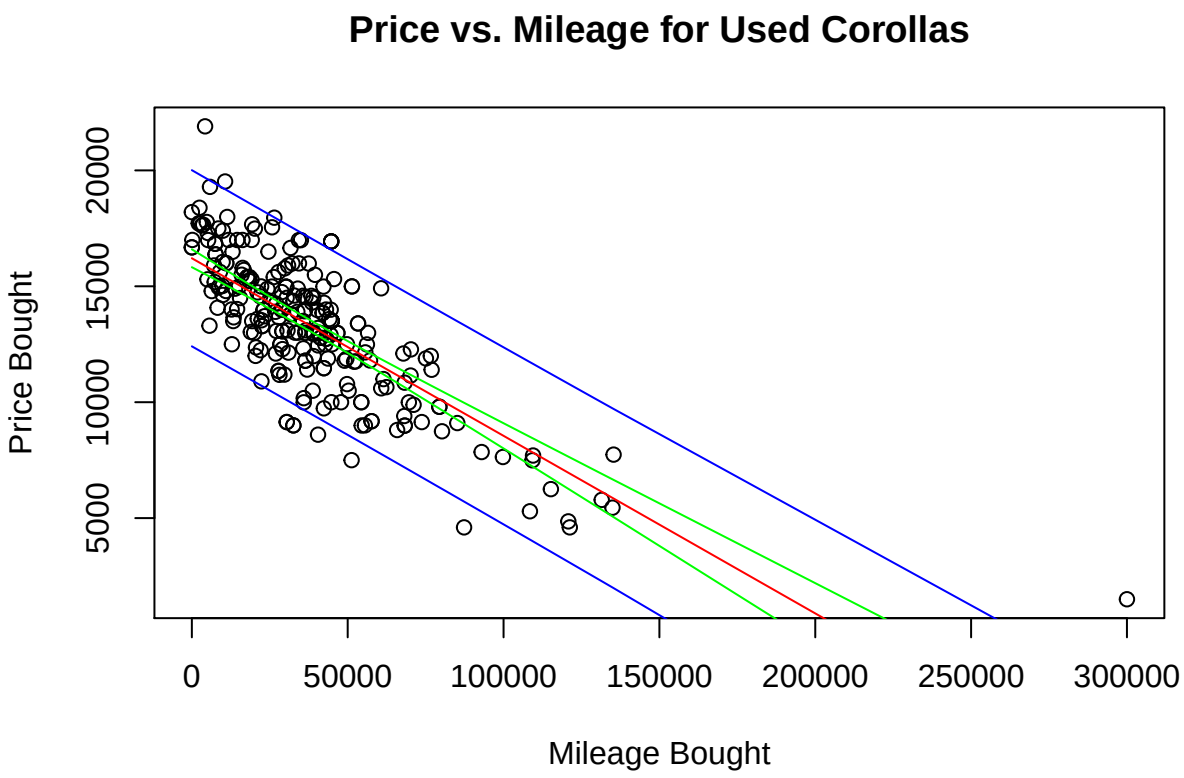
```r
# create confidence intervals for the betas
(CI_betas <- confint(lm_fit_all))
```

```
##                         2.5 %         97.5 %
## (Intercept)    15824.63955905 16595.87078104
## mileage_bought    -0.08450809    -0.06870578
```

```r
# create confidence interval for the regression line mu_y
CI_regression_line <- predict(lm_fit_all, interval="confidence", level = 0.95)

# create prediction interval for the regression line
prediction_interval <- predict(lm_fit_all, interval="predict")
```

```
## Warning in predict.lm(lm_fit_all, interval = "predict"): predictions on current data refer to _futur
```

```r
# plot both confidence interval and the prediction interval
plot(price_bought ~ mileage_bought, data = used_corollas_all,
     main = "Price vs. Mileage for Used Corollas", xlab = "Mileage Bought",
     ylab = "Price Bought")


# plot confidence interval
points(used_corollas_all$mileage_bought, CI_regression_line[, 1], col = "red", type = "l")
points(used_corollas_all$mileage_bought, CI_regression_line[, 2], col = "green", type = "l")
points(used_corollas_all$mileage_bought, CI_regression_line[, 3], col = "green", type = "l")


# plot prediction interval
points(used_corollas_all$mileage_bought, prediction_interval[, 2], col = "blue", type = "l")
points(used_corollas_all$mileage_bought, prediction_interval[, 3], col = "blue", type = "l")
```

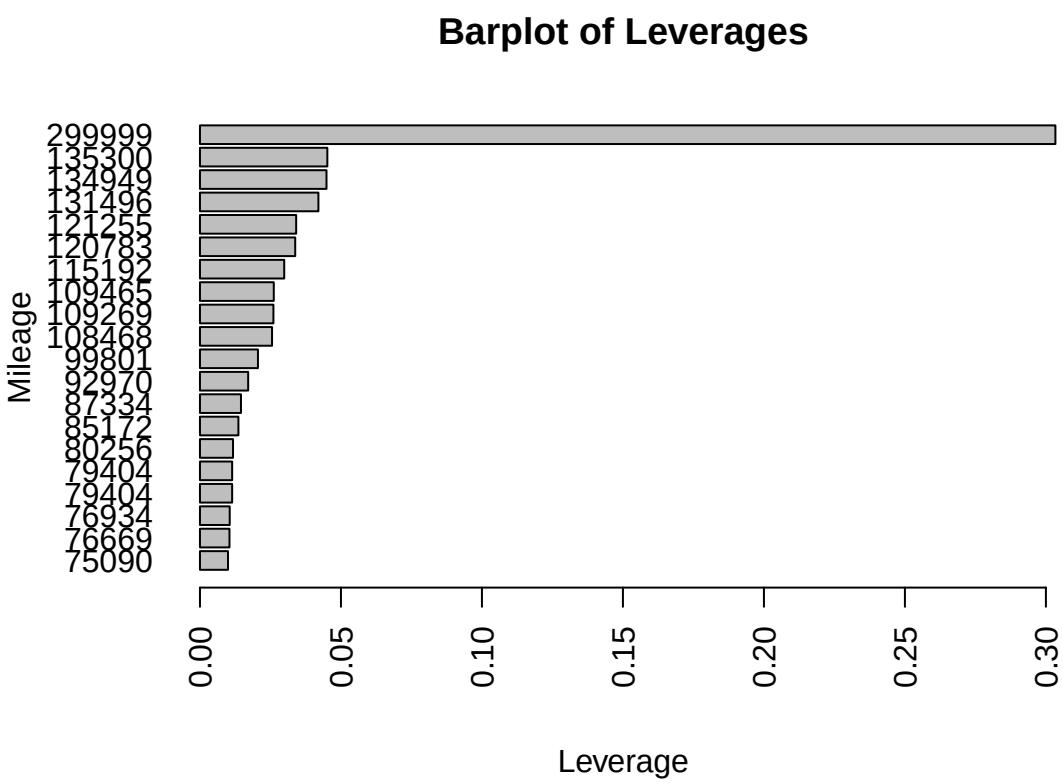


Answers:

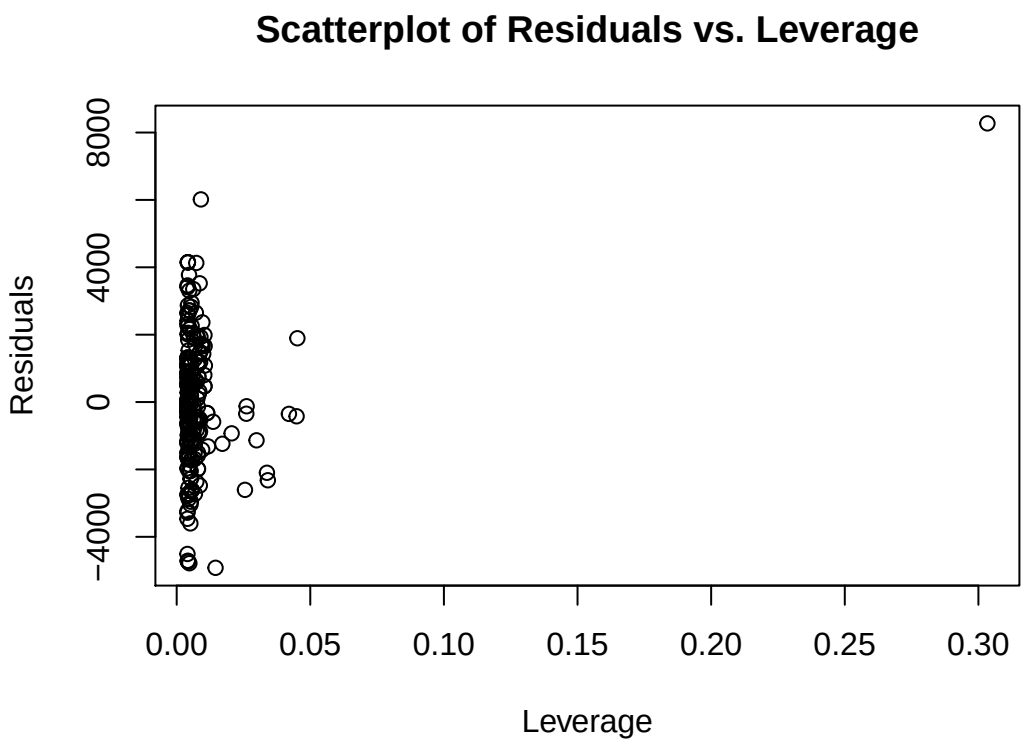
The equation for the confidence interval for the regression line is $\hat{y} \pm t^* \cdot SE_{\hat{\mu}}$, where $SE_{\hat{\mu}} = \sigma_{\in}\sqrt{\frac{1}{n} + \frac{(x^* - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}}$. The interval becomes wider for cars that have been driven the most miles because there is more uncertainty at the ends of the line. If we look at the equation we see that as x^*, our x-value, increases or decreases (getting further away from the mean $\bar{x}$), the standard error also increases as the numerator of the SE term gets larger, and therefore the confidence interval also gets wider.

Part 1.5 (10 points): Let's analyze the leverage and Cook's distance for the data points in the `used_corollas_all`. Calculate the leverage for the data points in this model (i.e., the hat values) and plot the 20 largest leverage values found using a bar plot. Also plot the residuals as a function of the leverage for each point, and use R's built in plot functions to plot Cook's distance for each data point and the standardized residuals as a function of the leverage for each point. Based on the 'rules of thumb' we discussed in class, how many points are considered 'very unusual' for the different measures of: Cook's distance, standardized residuals, studentized residuals, and leverage.

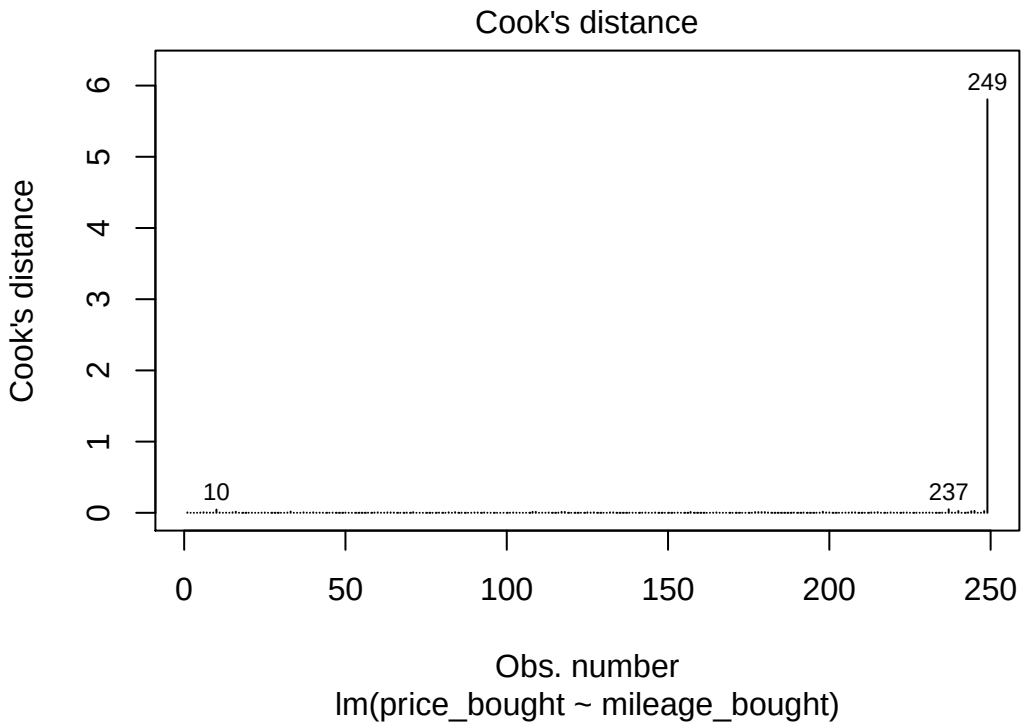
```
# get the h_i "hat values"
hat_vals <- hatvalues(lm_fit_all)
names(hat_vals) <- used_corollas_all$mileage_bought
par(mar = c(6, 6, 4, 4))
barplot(tail(hat_vals, 20), las=2, xlab = "Leverage", horiz = TRUE,
        ylab = "Mileage", mgp=c(4,1,0), main = "Barplot of Leverages")
```



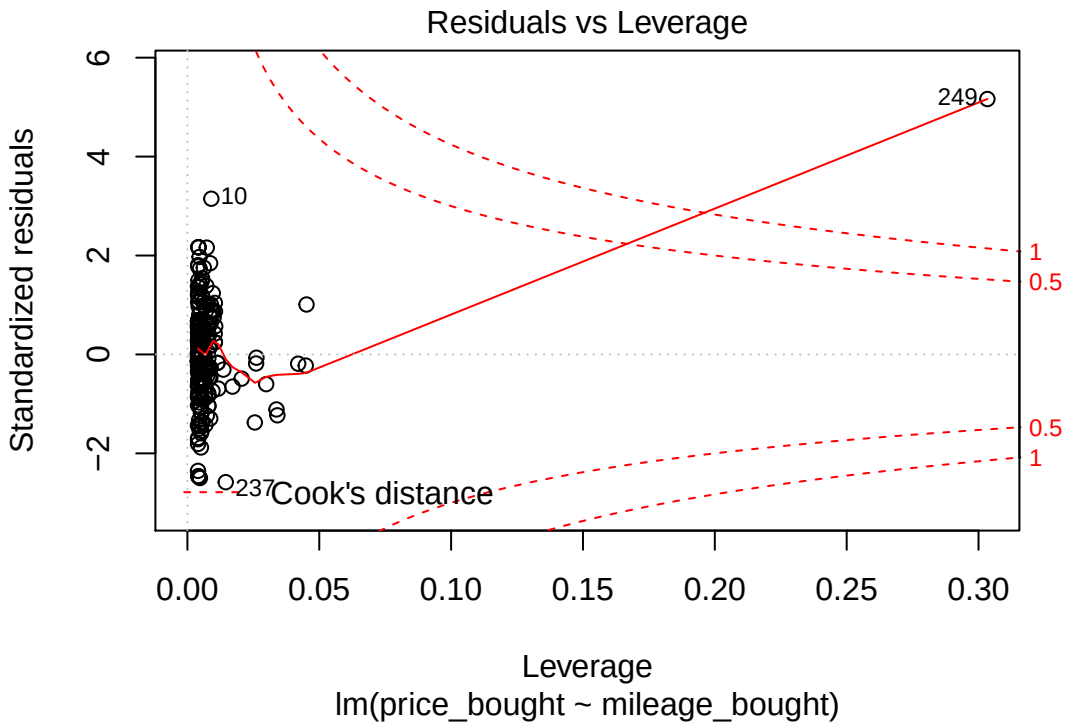

```r
# plot residuals as a function of the hat values
plot(hat_vals, lm_fit_all$residuals,
     xlab = "Leverage",
     ylab = "Residuals",
     main = "Scatterplot of Residuals vs. Leverage")
```




```
# plot Cook's distances and plot them
plot(lm_fit_all, 4)
```




```
# use the base R function to plot the standardized residuals with the hat values
#  along with contours that contain constant Cook's distance values
plot(lm_fit_all, 5)
```




```
# get the number of highly unusual points based on leverage, standardized residuals and studentized res

#Cook's distance:
sum(cooks.distance(lm_fit_all) > 1)
```

```
## [1] 1
```

```
#Leverage: above 3(k+1)/n where k = 1, so above 6/n
sum(hat_vals > 6/249)
```

```
## [1] 10
```

```
#Standardized Residuals: Beyond +/- 3
sum(rstandard(lm_fit_all) > 3, rstandard(lm_fit_all) < -3)
```

```
## [1] 2
```

```
#Studentized Residuals: Beyond +/- 3

sum(rstudent(lm_fit_all) > 3, rstandard(lm_fit_all) < -3)
```

```
## [1] 2
```

Answer

a) Cook's distance: 1 value where $D_i > 1$
b) Standardized Residuals: 2 values
c) Studentized Residuals: 2 values
d) Leverage: 10 values are very unusual

Part 1.6 (5 points): Above you fit two models: `lm_fit_all` which contained all the used Corollas and `lm_fit_150k` which did not contain the high leverage car with 300,000 miles driven. Describe below which you think is best and why? Also describe any limitation to these models.

Answer I think the `lm_fit_150k` model is better in this case because adding the one high leverage car changes the regression line too drastically. The predictions for 150000 miles are probably more accurate with the `lm_fit_150k` model since this number is close to the rest of our data points (and the 300000 mile car is much further away). The limitation to this model is that since the slope is steeper the x-intercept is going to be decreased, so it will become inaccurate faster as the mileage increases. If we are looking at cars with very high mileages perhaps the one including the 300000 miles-driven car would be better. However, of course `lm_fit_all` would be less accurate when looking at smaller mileages.

Part 2: Multiple linear regression

Let's now explore multiple linear regression where we try to predict a response variable y, based on several explanatory variables x_1, x_2 etc.

Part 2.1 (10 points): Let's start again by using dplyr to derive a new data set from the the original Edmunds `car_transactions` data set. Please create a data set in an object called `car_transactions2` that has the following properties:

1) It contains a new variable called `years_old` which is the difference between the year the car was sold, and the model year of the car.
2) It only contains used cars
3) It only contains the variables: price_bought, mileage_bought, years_old, msrp_bought

As a sanity check, if you have created this data frame correctly it should have 17,134 cases

Also, report what is the maximum and minimum value for the variable `years_old`. Explain why the minimum value of `years_old` makes sense (if the value doesn't make sense, read up about purchasing a new car and figure out what is going on). Finally, report what price the least and most expensive used cars sold for.

```
car_transactions <- car_transactions %>%  mutate(years_old = year_sold - model_year_bought)
car_transactions2 <- car_transactions %>% filter(new_or_used_bought == "U") %>% select(price_bought, mi

dim(car_transactions2)
```

```
## [1] 17134     4
```

```r
#Max and min of car "age"
max(car_transactions2$years_old)
```

```
## [1] 34
```

```r
min(car_transactions2$years_old)
```

```
## [1] -1
```

```r
#most and least expensive cars
max(na.omit(car_transactions2$price_bought))
```

```
## [1] 220000
```

```r
min(na.omit(car_transactions2$price_bought))
```

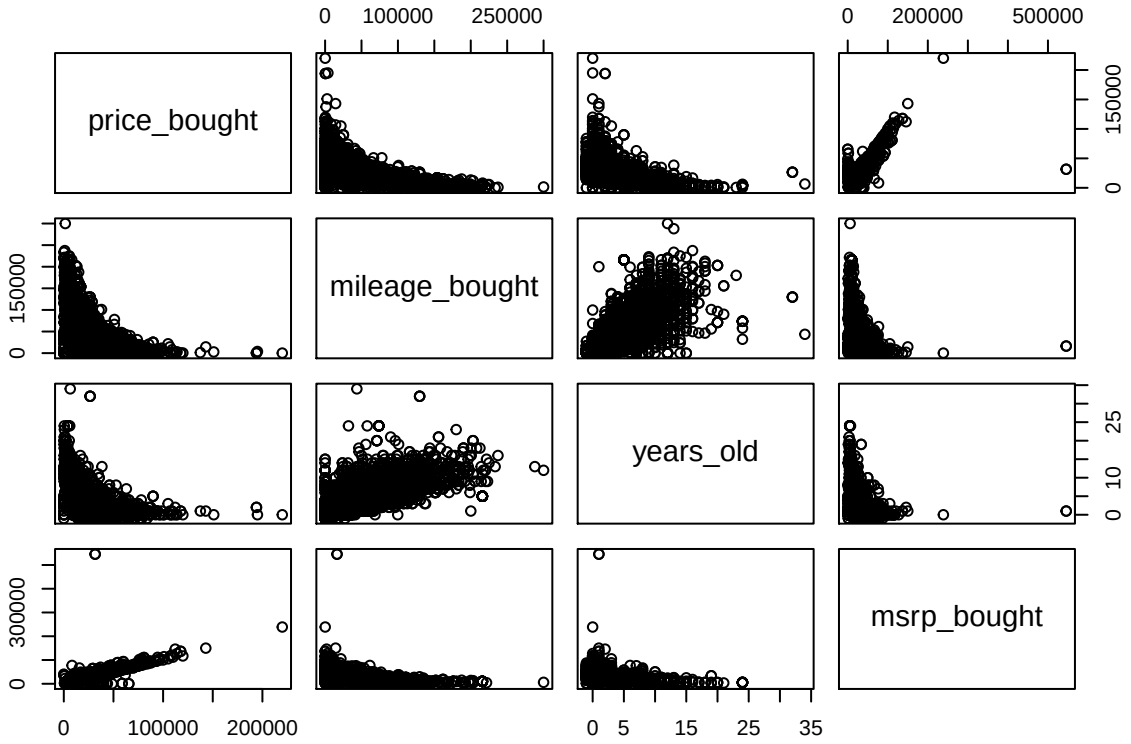
```
## [1] 0
```

Answer:

The maximum value for `years_old` is 34 while the minimum value is -1. This value still makes sense because sometimes car models for the following year are released the year before. For example, if someone could have bought a 2008 model in 2007 but then sold it the same year, making the car "-1" years old. The price of the most expensive car was 220000 and the price of the least expensive one was 0.

Part 2.2 (5 points): Now use the pairs() function to visualize the correlation between all pairs of variables in the `car_transactions2` data frame. Report whether any variable looks like it has a particiularly strong linear relationship with price_bought and whether it makes sense that there would be a strong relationship between these variables.

```r
pairs(car_transactions2)
```



Answer: There seems to be a pretty strong linear relationship between price_bought and msrp_bought. This makes sense because the original value of the car is going to determine the price it is sold for as a used car. A car that is originally a more expensive model will probably seel for more.

Part 2.3 (10 points): Next fit a multiple linear regression model predicting the price a car was bought for using the three variables mileage_bought, years_old, msrp_bought and save the linear fit to an object called `lm_cars`. Then use the summary() function to get information about the the linear regression model you fit by: a) saving the output of the summary() function to an object called `summary_lm_cars`, and b) print the output so you can see the result.

Report below the following information:

a) Do all the regression coefficients for all the variables appear to be statistically significant?
b) Do the signs for the regression coefficients make sense? Explain why.
c) Report what percentage of the total sum of squares is explained by this model by looking at the values stored in the `summary_lm_cars` object.

```
# fit the model
lm_cars <- lm(price_bought ~ mileage_bought + years_old + msrp_bought,
              data = car_transactions2)

# get information about the fit
(summary_lm_cars <- summary(lm_cars))
```

```
##
```

```
## Call:
## lm(formula = price_bought ~ mileage_bought + years_old + msrp_bought,
##     data = car_transactions2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -310279   -2934    -571    2238   63537
##
## Coefficients:
##                    Estimate   Std. Error t value            Pr(>|t|)
## (Intercept)    11155.428797   202.919846   54.98 <0.0000000000000002 ***
## mileage_bought    -0.046924     0.003563  -13.17 <0.0000000000000002 ***
## years_old       -447.222407    40.271434  -11.11 <0.0000000000000002 ***
## msrp_bought        0.608395     0.004918  123.71 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6759 on 8731 degrees of freedom
##   (8399 observations deleted due to missingness)
## Multiple R-squared:  0.7309, Adjusted R-squared:  0.7308
## F-statistic:  7904 on 3 and 8731 DF,  p-value: < 0.00000000000000022
```

```
# look at the percentage of variability explained
summary_lm_cars$r.squared
```

```
## [1] 0.730882
```

Answers:

a) The p-values for all of the regression coefficients are very small (close to 0), so it seems that they are all statistically significant.

b) The signs do all make sense. mileage_bough and year_old have negative signs, which makes sense because we expect the value of the car to decrease as it increases in mileage and gets older. On the other hand, msrp_bought should be positively correlated because the more the car is worth originally, the more it should be worth used.

c) The percentage of total sum squares explained by the model is 73.09% (the unadjusted R-squared value).

Part 2.4 (5 points): Now try to create a model that can account for as much of the variability in the y values by:

a) Using dplyr's mutate function to add new to variables that are derived from the variables in the `car_transactions2` data frame (i.e., square the variables, take logs, add interactions, etc). Save the new data frame to an object called `car_transactions3`.

b) Fit a linear model to the variables in this `car_transactions3`.

c) Calculate the R^2 value

d) Repeat this process to try to generate a R^2 value that is as large as possible, and report what this value is. Whoever get the largest R^2 value in the class will get a prize.

Then use LaTeX to write out the equation for the model you found and report whether you believe this is a good predictive model.

```
car_transactions3 <- car_transactions2 %>%
  mutate(x1 = log10(msrp_bought),
    x2 = years_old*x1,
    x3= mileage_bought*x1,
    x4= msrp_bought*years_old,
    x5 = msrp_bought*mileage_bought,
    x6 = msrp_bought^2,
    x7 = mileage_bought^2,
    x8 = years_old^2,
    x9 = (x6+ x7)^2,
    x10 = (x6 + x8)^2,
    x11 = x1*msrp_bought,
    x12 = x1^2,
    x13 = x12*years_old,
    x14 = x12*msrp_bought,
    x15 = x12*mileage_bought,
    x16 = x7*years_old,
    x17 = x7*msrp_bought,
    x18 = x7*mileage_bought)



lm_cars_3 <- lm(price_bought ~  mileage_bought + years_old + msrp_bought
                + x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 +
                  x11 + x12 + x13 +x14 +x15 + x16 + x17 + x18,
              data = car_transactions3)
summary(lm_cars_3)$r.squared
```

```
## [1] 0.9399118
```

```
summary(lm_cars_3)
```

```
##
## Call:
## lm(formula = price_bought ~ mileage_bought + years_old + msrp_bought +
##     x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11 +
##     x12 + x13 + x14 + x15 + x16 + x17 + x18, data = car_transactions3)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -44130  -1199    408   1602  27614
##
## Coefficients:
##                                   Estimate                  Std. Error
## (Intercept)    36298.418858623648702632636  2793.384579129906796879368
## mileage_bought    -1.656434271061017815541     0.181208096325387119085
```

```
## years_old      12782.194818159157875925303  1973.497454876552637870191
## msrp_bought     -155.102589349007018881821     8.960368929844099383786
## x1             18761.408659970911685377359  1269.701031223760082866647
## x2             -8421.369826823631228762679  1146.007343947455638044630
## x3                 0.445320361549493393127     0.108146097714992436845
## x4                -0.052158460999634941035     0.006778981127018952543
## x5                -0.000002871624610158920     0.000000710629260459196
## x6                 0.000025700593907413755     0.000001092797054444631
## x7                 0.000009938551611873941     0.000000283846638999070
## x8                 8.424957671312649054585     4.068653806679009221625
## x9                -0.000000000000000002597     0.000000000000000003008
## x10               -0.000000000000000024902     0.000000000000000003156
## x11               65.946310578620597198096     3.595839765828924949886
## x12            -3434.843276243590935337124   519.516110343833588558482
## x13             1311.621900396983619430102   168.672844391891146642593
## x14               -7.070266373064141518512     0.368601594572947888206
## x15               -0.009784282252063963500     0.016594284687419708774
## x16                0.000000013678142147541     0.000000002877972392517
## x17               -0.000000000058746379396     0.000000000005904905046
## x18               -0.000000000000795716588     0.000000000001451052561
##                t value             Pr(>|t|)
## (Intercept)     12.994 < 0.0000000000000002 ***
## mileage_bought  -9.141 < 0.0000000000000002 ***
## years_old        6.477  0.00000000009867742 ***
## msrp_bought    -17.310 < 0.0000000000000002 ***
## x1              14.776 < 0.0000000000000002 ***
## x2              -7.348  0.00000000000021862 ***
## x3               4.118  0.00003860947806059 ***
## x4              -7.694  0.00000000000001580 ***
## x5              -4.041  0.00005369032298564 ***
## x6              23.518 < 0.0000000000000002 ***
## x7               3.307             0.000948 ***
## x8               2.071             0.038416 *
## x9              -0.863             0.388008
## x10             -7.890  0.00000000000000338 ***
## x11             18.340 < 0.0000000000000002 ***
## x12             -6.612  0.00000000004024828 ***
## x13              7.776  0.00000000000000833 ***
## x14            -19.181 < 0.0000000000000002 ***
## x15             -0.590             0.555462
## x16              4.753  0.00000203931568374 ***
## x17             -9.949 < 0.0000000000000002 ***
## x18             -0.548             0.583451
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3197 on 8713 degrees of freedom
##   (8399 observations deleted due to missingness)
## Multiple R-squared:  0.9399, Adjusted R-squared:  0.9398
## F-statistic:  6490 on 21 and 8713 DF,  p-value: < 0.00000000000000022
```

Answer

We have $\hat{y}$ as the predicted price. The predictive model is $\hat{y} = 32987.45 + -1.823(mileage) + 12782.19(years) + -155.10(msrp) + 18761.40x_1 + -8421.36x_2 + 0.445x_3 + -0.0521x_4 + -0.00000287x_5 + 0.0000257x_6 +$

$0.000000938x_7 + 8.424x_8 + -0.000000000000000002x_9 + -0.000000000000000002x_{10} + 65.94x_{11} + -3434.84x_{12} + 1311.62x_{13} + -7.070x_{14} + -0.00978x_{15} + 0.00000001x_{16} + -0.0000000000587x_{17} + -0.000000000000795x_{18}$ (a long equation!). $x_1, \ldots x_{18}$ are defined in the above code. The R-squared value is 0.9399 which shows the model is a good predictor; however, there are certainly too many variables and there is probably an overfitting problem here, so it is probably not the best model.

Part 2.5 (2 points): If your model in part 2.4 is very large (i.e., has many variables) see if you can come up with a smaller model that captures a reasonable amount of the variability and describe whether you think this new model is better.

```
lm_cars_4 <- lm(price_bought ~  mileage_bought + years_old + msrp_bought
                + x1 + x6 ,
              data = car_transactions3)
summary(lm_cars_4)$r.squared
```

```
## [1] 0.9214954
```

Answer: Here we have far less variables but only a marginally smaller R-squared value, 0.9215. I would say this model is better since we don't has as much as an overfitting problem.

Reflection (3 points)

Please reflect on how the homework went by going to Canvas, going to the Quizzes link, and clicking on Reflection on homework 8